

Grammars and Topic Models

Mark Johnson

Department of Computing
Macquarie University

Outline

Topic models

Context-free grammars

Mixture topic models as PCFGs

Unsupervised learning of internal structure of named entities

Adaptor grammars: a Bayesian non-parametric extension to CFGs

Admixture topic models as PCFGs







Finding topical collocations with adaptor grammars

Conclusion and future work

Topic models for document processing







- Topic models *cluster documents into one or more topics*
 - ▶ usually *unsupervised* (i.e., topics aren't given in training data)
- Important for document analysis and information extraction
 - ▶ Example: clustering news stories for information retrieval
 - ▶ Example: tracking evolution of a research topic over time

Computers

 **US man pleads guilty in Sony data hack**
Ninemsn - 10 minutes ago     

A US college student who was a member of computer hacking group LulzSec has pleaded guilty to two federal charges of breaking into computers at Sony Pictures Entertainment. Cody Krestinger, 24, of Tempe, Arizona, entered his plea to one count each of ...

[Arizona college student pleads guilty to charges for hacking Sony Pictures ...](#) Washington Post
[Ariz. man pleads guilty in Sony data breach case](#) Newsday
[See all 95 sources >](#)

 **Half a million Mac computers 'infected with malware'**
BBC News - 10 hours ago     

More than half a million Apple computers have been infected with the Flashback Trojan, according to a Russian anti-virus firm.

[Mac Computers Affected by Hacker Attack: Researcher](#) BusinessWeek
[Apple Mac Computers Hit in Hacker Attack, Researcher Says](#) Bloomberg
[In Depth: Mac Botnet Infects More Than 600000 Apple Computers](#) eWeek
[See all 230 sources >](#)

Mixture versus admixture topic models

- In a *mixture model*, each document has a *single topic*

Mixture versus admixture topic models

- In a *mixture model*, each document has a *single topic*
 - ▶ all words in the document come from this topic

Mixture versus admixture topic models

- In a *mixture model*, each document has a *single topic*
 - ▶ all words in the document come from this topic
- In *admixture models*, each document has a *distribution over topics*

Mixture versus admixture topic models

- In a *mixture model*, each document has a *single topic*
 - ▶ all words in the document come from this topic
- In *admixture models*, each document has a *distribution over topics*
 - ▶ a single document can have multiple topics (number of topics in a document controlled by prior)

Mixture versus admixture topic models

- In a *mixture model*, each document has a *single topic*
 - ▶ all words in the document come from this topic
 - In *admixture models*, each document has a *distribution over topics*
 - ▶ a single document can have multiple topics (number of topics in a document controlled by prior)
- ⇒ can capture more complex relationships between documents than a mixture model

Mixture versus admixture topic models

- In a *mixture model*, each document has a *single topic*
 - ▶ all words in the document come from this topic
- In *admixture models*, each document has a *distribution over topics*
 - ▶ a single document can have multiple topics (number of topics in a document controlled by prior)
 - ⇒ can capture more complex relationships between documents than a mixture model
- Both mixture and admixture topic models typically use a “*bag of words*” representation of a document

Example: documents from NIPS corpus

Annotating an unlabeled dataset is one of the bottlenecks in using supervised learning to build good predictive models. Getting a dataset labeled by experts can be expensive and time consuming. With the advent of crowdsourcing services ...

The task of recovering intrinsic images is to separate a given input image into its material-dependent properties, known as reflectance or albedo, and its light-dependent properties, such as shading, shadows, specular highlights, ...

In each trial of a standard visual short-term memory experiment, subjects are first presented with a display containing multiple items with simple features (e.g. colored squares) for a brief duration and then, after a delay interval, their memory for ...

Many studies have uncovered evidence that visual cortex contains specialized regions involved in processing faces but not other object classes. Recent electrophysiology studies of cells in several of these specialized regions revealed that at least some ...

Example (cont): ignore function words

Annotating an unlabeled dataset is one of the bottlenecks in using supervised learning to build good predictive models. Getting a dataset labeled by experts can be expensive and time consuming. With the advent of crowdsourcing services ...

The task of recovering intrinsic images is to separate a given input image into its material-dependent properties, known as reflectance or albedo, and its light-dependent properties, such as shading, shadows, specular highlights, ...

In each trial of a standard visual short-term memory experiment, subjects are first presented with a display containing multiple items with simple features (e.g. colored squares) for a brief duration and then, after a delay interval, their memory for ...

Many studies have uncovered evidence that visual cortex contains specialized regions involved in processing faces but not other object classes. Recent electrophysiology studies of cells in several of these specialized regions revealed that at least some ...

Example (cont): mixture topic model

Annotating an unlabeled dataset is one of the bottlenecks in using supervised learning to build good predictive models. Getting a dataset labeled by experts can be expensive and time consuming. With the advent of crowdsourcing services ...

The task of recovering intrinsic images is to separate a given input image into its material-dependent properties, known as reflectance or albedo, and its light-dependent properties, such as shading, shadows, specular highlights, ...

In each trial of a standard visual short-term memory experiment, subjects are first presented with a display containing multiple items with simple features (e.g. colored squares) for a brief duration and then, after a delay interval, their memory for ...

Many studies have uncovered evidence that visual cortex contains specialized regions involved in processing faces but not other object classes. Recent electrophysiology studies of cells in several of these specialized regions revealed that at least some ...

Example (cont): admixture topic model

Annotating an unlabeled dataset is one of the bottlenecks in using supervised learning to build good predictive models. Getting a dataset labeled by experts can be expensive and time consuming. With the advent of crowdsourcing services ...

The task of recovering intrinsic images is to separate a given input image into its material-dependent properties, known as reflectance or albedo, and its light-dependent properties, such as shading, shadows, specular highlights, ...

In each trial of a standard visual short-term memory experiment, subjects are first presented with a display containing multiple items with simple features (e.g. colored squares) for a brief duration and then, after a delay interval, their memory for ...

Many studies have uncovered evidence that visual cortex contains specialized regions involved in processing faces but not other object classes. Recent electrophysiology studies of cells in several of these specialized regions revealed that at least some ...

My contribution: collocation topic models

Annotating an unlabeled dataset is one of the bottlenecks in using supervised learning to build good predictive models. Getting a dataset labeled by experts can be expensive and time consuming. With the advent of crowdsourcing services ...

The task of recovering intrinsic images is to separate a given input image into its material-dependent properties, known as reflectance or albedo, and its light-dependent properties, such as shading, shadows, specular highlights, ...

In each trial of a standard visual short-term memory experiment, subjects are first presented with a display containing multiple items with simple features (e.g. colored squares) for a brief duration and then, after a delay interval, their memory for ...

Many studies have uncovered evidence that visual cortex contains specialized regions involved in processing faces but not other object classes. Recent electrophysiology studies of cells in several of these specialized regions revealed that at least some ...

Mixture versus admixture models

- Admixture models are more complex than mixture models
 - ⇒ Admixture models often require more data to learn
- Mixture models can describe shorter documents (phrases or clauses) fairly well, where one topic per document assumption is not too bad
- Admixture models are better for longer documents, which are likely to have more than one topic

Problems with the “bag of words” assumption

- Each word is *generated independently* given the document’s topic(s)
- ⇒ Ignores relationship between adjacent words
- But especially in technical documents, much specialist terminology consists of *multi-word expressions* or *collocations*
 - ▶ *membrane potential* is strongly indicative of NEUROSCIENCE, but *membrane* and *potential* are not
 - ▶ *neural network* is strongly indicative of MACHINE LEARNING, but *neural* and *network* are not
- Topic models can *identify important terminology of a field*
 - ▶ much more useful when *multi-word expressions* are also identified

Using Bayesian inference to find document topics and topical words

- Define *generative models* M of *document collections* \mathcal{D}
 - ▶ $P(\mathcal{D} | M)$ is probability of generating \mathcal{D} from M
- Define a *Bayesian prior* $P(M)$ over possible generative models M
 - ▶ $P(M)$ is chosen to prefer “simpler” models
- Goal: find the highest probability model M given document collection \mathcal{D}
- Bayes rule “inverts” the generative process:

$$\underbrace{P(M | \mathcal{D})}_{\text{Posterior}} \propto \underbrace{P(\mathcal{D} | M)}_{\text{Likelihood}} \underbrace{P(M)}_{\text{Prior}}$$

- Computational challenge: find models M with high posterior probability

Generative mixture and admixture topic models

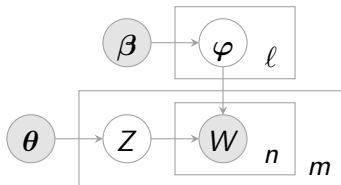
- In *mixture models*, each document has a *single topic*
 - ▶ **Generative model:**
 - for each topic i , generate a *word distribution* φ_i
 - for each document j
 - generate a *document topic* z_j from set of topics
 - for each word position k , generate *word* w_k
 - from document topic's word distribution φ_{z_j}
- In *admixture models*, each document has a *distribution over topics*
 - ▶ **Generative model:**
 - for each topic i , generate a *word distribution* φ_i
 - for each document j
 - generate a *document topic distribution* θ_j over topics
 - for each word position k
 - generate *word's topic* z_k from document's topic distribution θ_j
 - generate *word* w_k from word's topic distribution φ_{z_k}

Bayesian inference for topic models

- Both mixture and admixture topic models are *products of multinomial distributions*
 - ⇒ conjugate prior is a product of *Dirichlet distributions*
- Dirichlet prior can bias towards *sparse distributions*
 - ▶ few words in each topic
 - ▶ few topics in each document (admixture model)
- Standard Bayesian inference procedures can be used to learn topic models
 - ▶ Markov Chain Monte Carlo (MCMC)
 - ▶ Mean-field Variational Bayes

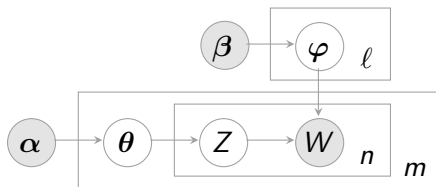
Mixture topic model (formal description)

$$\begin{array}{l|l|l} \varphi_i & \beta & \sim \text{Dir}(\beta) \quad i = 1, \dots, \ell = \text{number of topics} \\ z_j & \theta & \sim \text{Cat}(\theta) \quad j = 1, \dots, m = \text{number of documents} \\ w_{j,k} & \varphi, z_j & \sim \varphi_{z_j} \quad j = 1, \dots, m \\ & & k = 1, \dots, n = \text{number of words in document } j \end{array}$$



Admixture (LDA) topic model (formal description)

$$\begin{aligned}\varphi_i &\sim \text{Dir}(\boldsymbol{\beta}) & i = 1, \dots, \ell = \text{number of topics} \\ \boldsymbol{\theta}_j &\sim \text{Dir}(\boldsymbol{\alpha}) & j = 1, \dots, m = \text{number of documents} \\ z_{j,k} &\sim \text{Cat}(\boldsymbol{\theta}_j) & j = 1, \dots, m \\ & & k = 1, \dots, n = \text{number of words in a document} \\ w_{j,k} &\sim \text{Cat}(\varphi_{z_{j,k}}) & j = 1, \dots, m \\ & & k = 1, \dots, n\end{aligned}$$



Outline

Topic models

Context-free grammars

Mixture topic models as PCFGs

Unsupervised learning of internal structure of named entities

Adaptor grammars: a Bayesian non-parametric extension to CFGs

Admixture topic models as PCFGs

Finding topical collocations with adaptor grammars

Conclusion and future work

Grammars and automata

- Context-Free Grammars (CFGs) were initially developed as a formal model of hierarchical constituent structure in natural languages
- But they are useful for other applications as well
 - ▶ important tool for compiler design
 - ▶ strong connections with push-down automata
- Here we use Probabilistic CFGs as compact specifications of stochastic automata

Probabilistic context-free grammars

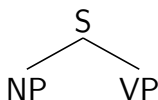
- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
 - ▶ choosing a rule expanding that nonterminal, and
 - ▶ recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

<i>Probability θ_r</i>	<i>Rule r</i>	
1	$S \rightarrow NP VP$	S
0.7	$NP \rightarrow Sam$	
0.3	$NP \rightarrow Sandy$	
1	$VP \rightarrow V NP$	
0.8	$V \rightarrow likes$	
0.2	$V \rightarrow hates$	

Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
 - ▶ choosing a rule expanding that nonterminal, and
 - ▶ recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

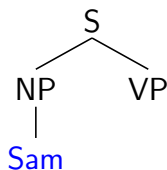
<i>Probability θ_r</i>	<i>Rule r</i>
1	$S \rightarrow NP VP$
0.7	$NP \rightarrow Sam$
0.3	$NP \rightarrow Sandy$
1	$VP \rightarrow V NP$
0.8	$V \rightarrow likes$
0.2	$V \rightarrow hates$



Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
 - ▶ choosing a rule expanding that nonterminal, and
 - ▶ recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

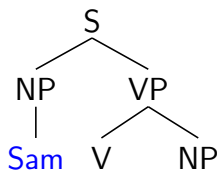
<i>Probability θ_r</i>	<i>Rule r</i>
1	$S \rightarrow NP VP$
0.7	$NP \rightarrow Sam$
0.3	$NP \rightarrow Sandy$
1	$VP \rightarrow V NP$
0.8	$V \rightarrow likes$
0.2	$V \rightarrow hates$



Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
 - ▶ choosing a rule expanding that nonterminal, and
 - ▶ recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

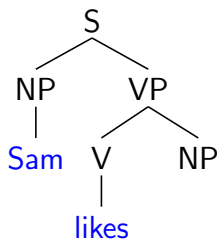
<i>Probability θ_r</i>	<i>Rule r</i>
1	$S \rightarrow NP VP$
0.7	$NP \rightarrow Sam$
0.3	$NP \rightarrow Sandy$
1	$VP \rightarrow V NP$
0.8	$V \rightarrow likes$
0.2	$V \rightarrow hates$



Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
 - ▶ choosing a rule expanding that nonterminal, and
 - ▶ recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

<i>Probability θ_r</i>	<i>Rule r</i>
1	$S \rightarrow NP VP$
0.7	$NP \rightarrow Sam$
0.3	$NP \rightarrow Sandy$
1	$VP \rightarrow V NP$
0.8	$V \rightarrow likes$
0.2	$V \rightarrow hates$

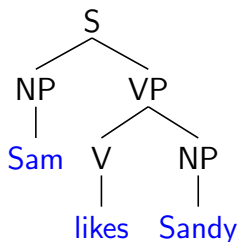


$$P(\text{Tree}) = 1 \times 0.7 \times 1 \times 0.8 \times$$

Probabilistic context-free grammars

- Probabilistic context-free grammars (PCFGs) define *probability distributions over trees*
- Each *nonterminal node* expands by
 - ▶ choosing a rule expanding that nonterminal, and
 - ▶ recursively expanding any nonterminal children it contains
- Probability of tree is *product of probabilities of rules* used to construct it

<i>Probability θ_r</i>	<i>Rule r</i>
1	$S \rightarrow NP VP$
0.7	$NP \rightarrow Sam$
0.3	$NP \rightarrow Sandy$
1	$VP \rightarrow V NP$
0.8	$V \rightarrow likes$
0.2	$V \rightarrow hates$



$$P(\text{Tree}) = 1 \times 0.7 \times 1 \times 0.8 \times 0.3$$

Context-Free Grammars (formal definition)

- A CFG $(\mathcal{N}, \mathcal{W}, \mathcal{R}, S)$ defines *sets of trees* \mathcal{T}_X for each $X \in \mathcal{N} \cup \mathcal{W}$:
 - ▶ if $X \in \mathcal{W}$ then $\mathcal{T}_X = \{X\}$ (the 1-node tree labelled X)
 - ▶ if $X \in \mathcal{N}$ then:

$$\mathcal{T}_X = \bigcup_{X \rightarrow B_1 \dots B_n \in \mathcal{R}_X} \text{TREE}_X(\mathcal{T}_{B_1}, \dots, \mathcal{T}_{B_n})$$

where $\mathcal{R}_A = \{A \rightarrow \beta : A \rightarrow \beta \in \mathcal{R}\}$ for each $A \in \mathcal{N}$, and

$$\text{TREE}_A(\mathcal{T}_{B_1}, \dots, \mathcal{T}_{B_n}) = \left\{ \begin{array}{c} A \\ \wedge \\ t_1 \dots t_n \end{array} : \begin{array}{l} t_i \in \mathcal{T}_{B_i}, \\ i = 1, \dots, n \end{array} \right\}$$

That is, $\text{TREE}_A(\mathcal{T}_{B_1}, \dots, \mathcal{T}_{B_n})$ consists of the set of trees with whose root node is labelled A and whose i th child is a member of \mathcal{T}_{B_i} .

Probabilistic Context-Free Grammars

- A PCFG is a CFG $(\mathcal{N}, \mathcal{W}, \mathcal{R}, S)$ and multinomials θ_A over R_A for each $A \in \mathcal{N}$
 - ▶ $\theta_{A \rightarrow \beta}$ is the probability of A expanding to β
- A PCFG associates each $X \in \mathcal{N} \cup \mathcal{W}$ with a *distribution G_X over trees \mathcal{T}_X*
 - ▶ if $X \in \mathcal{W}$ then $G_X(X) = 1$ (i.e., all mass concentrated on 1-node tree)
 - ▶ if $A \in \mathcal{N}$ then:

$$G_A(t) = \sum_{A \rightarrow B_1 \dots B_n \in R_A} \theta_{A \rightarrow B_1 \dots B_n} \text{TD}_A(G_{B_1}, \dots, G_{B_n})(t) \quad (1)$$

where:

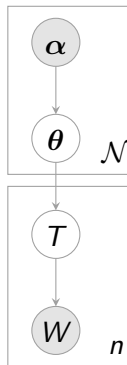
$$\text{TD}_A(G_1, \dots, G_n) \left(\begin{array}{c} A \\ \wedge \\ t_1 \dots t_n \end{array} \right) = \prod_{i=1}^n G_i(t_i).$$

That is, $\text{TD}_A(G_1, \dots, G_n)$ is a distribution over \mathcal{T}_A where each subtree t_i is generated independently from G_i .

Bayesian inference for PCFGs

- Each rule's probability $\theta_{A \rightarrow \beta}$ is given by a *category distribution* associated with its parent A
- ⇒ A PCFG can be viewed as a *product of multinomials*
- The conjugate prior is a *product of Dirichlet distributions* (one per nonterminal)
 - Bayesian inference procedures for PCFGs
 - ▶ Markov Chain Monte Carlo (Metropolis-within-Gibbs)
 - ▶ Mean-field variational Bayes

$$\begin{array}{l|l} \boldsymbol{\theta}_A & \boldsymbol{\alpha}_A \sim \text{Dir}(\boldsymbol{\alpha}_A), \quad A \in \mathcal{N} = \text{set of nonterminals} \\ T_i & \boldsymbol{\theta}_A \sim \text{TD}_S(\boldsymbol{\theta}), \quad i = 1, \dots, n = \text{number of sentences} \\ \mathbf{W}_i & T_i = \text{Yield}(T_i). \quad \mathbf{W}_i \text{ is sequence of words in sentence } i \end{array}$$



Outline

Topic models

Context-free grammars

Mixture topic models as PCFGs

Unsupervised learning of internal structure of named entities

Adaptor grammars: a Bayesian non-parametric extension to CFGs

Admixture topic models as PCFGs

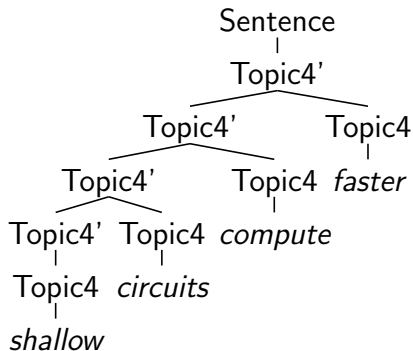
Finding topical collocations with adaptor grammars

Conclusion and future work

Mixture topic models as PCFGs (1)

- Idea: Design PCFG so that:
 - ▶ non-deterministic rules implement generative steps in topic model
 - ▶ deterministic rules propagate information to appropriate place

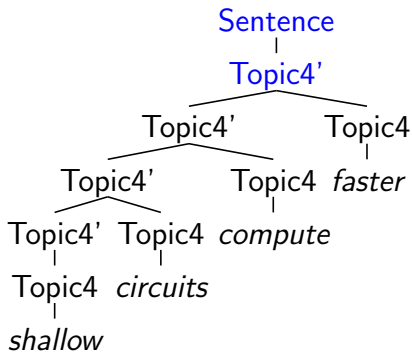
Sentence \rightarrow Topic'_{*i*} $i \in 1, \dots, \ell$
Topic'_{*i*} \rightarrow Topic'_{*i*} Topic_{*i*} $i \in 1, \dots, \ell$
Topic'_{*i*} \rightarrow Topic_{*i*} $i \in 1, \dots, \ell$
Topic_{*i*} \rightarrow w $i \in 1, \dots, \ell$
 $w \in \mathcal{W}$



Mixture topic models as PCFGs (2)

- Choose a topic for sentence (non-deterministically)

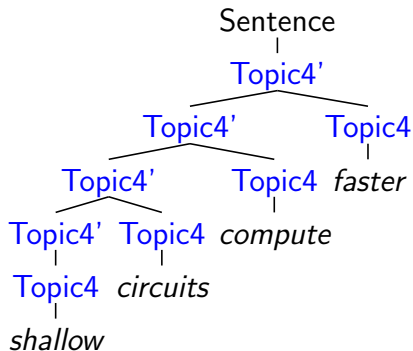
Sentence \rightarrow Topic'_{*i*} $i \in 1, \dots, \ell$
Topic'_{*i*} \rightarrow Topic'_{*i*} Topic_{*i*} $i \in 1, \dots, \ell$
Topic'_{*i*} \rightarrow Topic_{*i*} $i \in 1, \dots, \ell$
Topic_{*i*} \rightarrow w $i \in 1, \dots, \ell$
 $w \in \mathcal{W}$



Mixture topic models as PCFGs (3)

- Copy sentence topic to each word (deterministically)

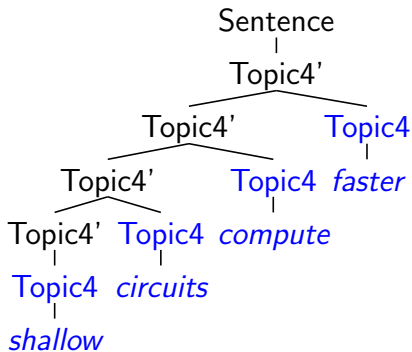
Sentence \rightarrow Topic'_{*i*} $i \in 1, \dots, \ell$
Topic'_{*i*} \rightarrow Topic'_{*i*} Topic_{*i*} $i \in 1, \dots, \ell$
Topic'_{*i*} \rightarrow Topic_{*i*} $i \in 1, \dots, \ell$
Topic_{*i*} \rightarrow w $i \in 1, \dots, \ell$
 $w \in \mathcal{W}$



Mixture topic models as PCFGs (4)

- Generate each word from sentence topic (non-deterministically)

Sentence \rightarrow Topic $'_i$ $i \in 1, \dots, \ell$
Topic $'_i \rightarrow$ Topic $'_i$ Topic $_i$ $i \in 1, \dots, \ell$
Topic $'_i \rightarrow$ Topic $_i$ $i \in 1, \dots, \ell$
Topic $_i \rightarrow w$ $i \in 1, \dots, \ell$
 $w \in \mathcal{W}$



Why are these reductions interesting?

- *Not* claiming that topic modelling should be done using PCFGs

Why are these reductions interesting?

- *Not* claiming that topic modelling should be done using PCFGs
 - ▶ PCFG parsing takes time proportional to *cube* of document length

Why are these reductions interesting?

- *Not* claiming that topic modelling should be done using PCFGs
 - ▶ PCFG parsing takes time proportional to *cube* of document length
 - ▶ standard topic model algorithms take time *linear* in document length

Why are these reductions interesting?

- *Not* claiming that topic modelling should be done using PCFGs
 - ▶ PCFG parsing takes time proportional to *cube* of document length
 - ▶ standard topic model algorithms take time *linear* in document length
- The PCFG reductions suggest *new kinds of models that merge grammars and topic models*

Why are these reductions interesting?

- *Not* claiming that topic modelling should be done using PCFGs
 - ▶ PCFG parsing takes time proportional to *cube* of document length
 - ▶ standard topic model algorithms take time *linear* in document length
- The PCFG reductions suggest *new kinds of models that merge grammars and topic models*
 - ▶ easily implemented and evaluated on small corpora

Why are these reductions interesting?

- *Not* claiming that topic modelling should be done using PCFGs
 - ▶ PCFG parsing takes time proportional to *cube* of document length
 - ▶ standard topic model algorithms take time *linear* in document length
- The PCFG reductions suggest *new kinds of models that merge grammars and topic models*
 - ▶ easily implemented and evaluated on small corpora
- Grammars are good at:

Why are these reductions interesting?

- *Not* claiming that topic modelling should be done using PCFGs
 - ▶ PCFG parsing takes time proportional to *cube* of document length
 - ▶ standard topic model algorithms take time *linear* in document length
- The PCFG reductions suggest *new kinds of models that merge grammars and topic models*
 - ▶ easily implemented and evaluated on small corpora
- Grammars are good at:
 - ▶ grouping words into hierarchically-structured larger units

Why are these reductions interesting?

- *Not* claiming that topic modelling should be done using PCFGs
 - ▶ PCFG parsing takes time proportional to *cube* of document length
 - ▶ standard topic model algorithms take time *linear* in document length
- The PCFG reductions suggest *new kinds of models that merge grammars and topic models*
 - ▶ easily implemented and evaluated on small corpora
- Grammars are good at:
 - ▶ grouping words into hierarchically-structured larger units
 - ▶ tracking relative ordering of these units

Outline

Topic models

Context-free grammars

Mixture topic models as PCFGs

Unsupervised learning of internal structure of named entities

Adaptor grammars: a Bayesian non-parametric extension to CFGs

Admixture topic models as PCFGs

Finding topical collocations with adaptor grammars

Conclusion and future work

The internal structure of named entities

- Named entities: people, companies, places, etc.

The internal structure of named entities

- Named entities: people, companies, places, etc.
- Represented as flat Noun Phrases (NPs) in Penn WSJ treebank

The internal structure of named entities

- Named entities: people, companies, places, etc.
- Represented as flat Noun Phrases (NPs) in Penn WSJ treebank
- Internal structure useful for e.g. coreference resolution

The internal structure of named entities

- Named entities: people, companies, places, etc.
- Represented as flat Noun Phrases (NPs) in Penn WSJ treebank
- Internal structure useful for e.g. coreference resolution
 - ▶ *Bill Clinton* and *Hillary Clinton* are unlikely to corefer because *Bill* and *Hillary* are both first names

The internal structure of named entities

- Named entities: people, companies, places, etc.
- Represented as flat Noun Phrases (NPs) in Penn WSJ treebank
- Internal structure useful for e.g. coreference resolution
 - ▶ *Bill Clinton* and *Hillary Clinton* are unlikely to corefer because *Bill* and *Hillary* are both first names
 - ▶ *Secretary Clinton* and *Hillary Clinton* can corefer because *Secretary* is an honorific

The internal structure of named entities

- Named entities: people, companies, places, etc.
- Represented as flat Noun Phrases (NPs) in Penn WSJ treebank
- Internal structure useful for e.g. coreference resolution
 - ▶ *Bill Clinton* and *Hillary Clinton* are unlikely to corefer because *Bill* and *Hillary* are both first names
 - ▶ *Secretary Clinton* and *Hillary Clinton* can corefer because *Secretary* is an honorific
- *Can we learn the internal structure of named entities?*

The internal structure of named entities

- Named entities: people, companies, places, etc.
- Represented as flat Noun Phrases (NPs) in Penn WSJ treebank
- Internal structure useful for e.g. coreference resolution
 - ▶ *Bill Clinton* and *Hillary Clinton* are unlikely to corefer because *Bill* and *Hillary* are both first names
 - ▶ *Secretary Clinton* and *Hillary Clinton* can corefer because *Secretary* is an honorific
- *Can we learn the internal structure of named entities?*
 - ▶ document ~ base NP consisting of proper nouns
 - ▶ topic ~ type of named entity

The internal structure of named entities

- Named entities: people, companies, places, etc.
- Represented as flat Noun Phrases (NPs) in Penn WSJ treebank
- Internal structure useful for e.g. coreference resolution
 - ▶ *Bill Clinton* and *Hillary Clinton* are unlikely to corefer because *Bill* and *Hillary* are both first names
 - ▶ *Secretary Clinton* and *Hillary Clinton* can corefer because *Secretary* is an honorific
- *Can we learn the internal structure of named entities?*
 - ▶ document ~ base NP consisting of proper nouns
 - ▶ topic ~ type of named entity
 - ▶ two “topics” (person names, company names)

The internal structure of named entities

- Named entities: people, companies, places, etc.
- Represented as flat Noun Phrases (NPs) in Penn WSJ treebank
- Internal structure useful for e.g. coreference resolution
 - ▶ *Bill Clinton* and *Hillary Clinton* are unlikely to corefer because *Bill* and *Hillary* are both first names
 - ▶ *Secretary Clinton* and *Hillary Clinton* can corefer because *Secretary* is an honorific
- *Can we learn the internal structure of named entities?*
 - ▶ document ~ base NP consisting of proper nouns
 - ▶ topic ~ type of named entity
 - ▶ two “topics” (person names, company names)
 - ▶ each “topic” has *six ordered positions*

The internal structure of named entities

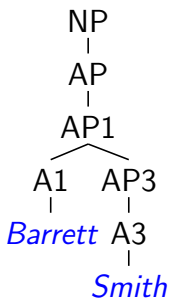
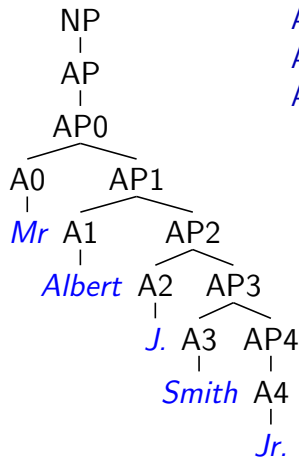
- Named entities: people, companies, places, etc.
- Represented as flat Noun Phrases (NPs) in Penn WSJ treebank
- Internal structure useful for e.g. coreference resolution
 - ▶ *Bill Clinton* and *Hillary Clinton* are unlikely to corefer because *Bill* and *Hillary* are both first names
 - ▶ *Secretary Clinton* and *Hillary Clinton* can corefer because *Secretary* is an honorific
- *Can we learn the internal structure of named entities?*
 - ▶ document ~ base NP consisting of proper nouns
 - ▶ topic ~ type of named entity
 - ▶ two “topics” (person names, company names)
 - ▶ each “topic” has *six ordered positions*
 - ▶ learn which words occur in which position of each “topic”

The internal structure of named entities

- Named entities: people, companies, places, etc.
- Represented as flat Noun Phrases (NPs) in Penn WSJ treebank
- Internal structure useful for e.g. coreference resolution
 - ▶ *Bill Clinton* and *Hillary Clinton* are unlikely to corefer because *Bill* and *Hillary* are both first names
 - ▶ *Secretary Clinton* and *Hillary Clinton* can corefer because *Secretary* is an honorific
- *Can we learn the internal structure of named entities?*
 - ▶ document ~ base NP consisting of proper nouns
 - ▶ topic ~ type of named entity
 - ▶ two “topics” (person names, company names)
 - ▶ each “topic” has *six ordered positions*
 - ▶ learn which words occur in which position of each “topic”
- Used in *unsupervised cross-document coreference model* of Elsner, Charniak and Johnson (2009)

A mixture topic grammar for named entities

$NP \rightarrow AP$ $NP \rightarrow BP$
 $AP \rightarrow AP_i$ $BP \rightarrow BP_i$ $0 \leq i \leq 6$
 $AP_i \rightarrow A_i$ $BP_i \rightarrow B_i$ $0 \leq i \leq 6$
 $AP_i \rightarrow A_i AP_j$ $BP_i \rightarrow B_i BP_j$ $0 \leq i < j \leq 6$



Analyses of company names

NP \rightarrow AP

NP \rightarrow BP

AP \rightarrow AP_{*i*}

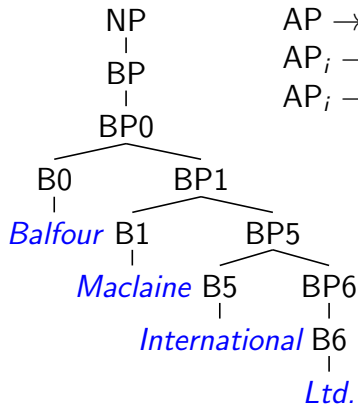
BP \rightarrow BP_{*i*} $0 \leq i \leq 6$

AP_{*i*} \rightarrow A_{*i*}

BP_{*i*} \rightarrow B_{*i*} $0 \leq i \leq 6$

AP_{*i*} \rightarrow A_{*i*} AP_{*j*}

BP_{*i*} \rightarrow B_{*i*} BP_{*j*} $0 \leq i < j \leq 6$



Outline

Topic models

Context-free grammars

Mixture topic models as PCFGs

Unsupervised learning of internal structure of named entities

Adaptor grammars: a Bayesian non-parametric extension to CFGs

Admixture topic models as PCFGs

Finding topical collocations with adaptor grammars

Conclusion and future work

From Multinomials to Dirichlet Processes

- Dirichlet Processes (DPs) are the *infinite-dimensional generalisation* of Dirichlet-Multinomials
- *Predictive distribution*: predict z_{n+1} given observations $\mathbf{z} = (z_1, \dots, z_n)$
 - ▶ *Finite set* of outcomes $(1, \dots, m)$:
Dirichlet-multinomial with prior $\alpha = (\alpha_1, \dots, \alpha_m)$

$$P(Z_{n+1} = k \mid \mathbf{z}) \propto n_k(\mathbf{z}) + \alpha_k$$

where $n_k(\mathbf{z})$ is the number of times k appears in $\mathbf{z} = (z_1, \dots, z_n)$

- ▶ *Infinite set* of outcomes Ω :
Dirichlet process $DP(\alpha, P_0)$ with *base distribution* $P_0(Z)$ and *concentration parameter* α

$$P(Z_{n+1} = z' \mid \mathbf{z}) \propto n_{z'}(\mathbf{z}) + \alpha P_0(z')$$

Dirichlet Processes as Adaptors

- DPs generalise Dirichlet-multinomials

$$P(Z_{n+1} = z' | \mathbf{z}) \propto n_{z'}(\mathbf{z}) + \alpha P_0(z')$$

- DPs follow a “*rich get richer*” law
 - ▶ frequent outcomes are increasingly likely to be predicted
- The DP is *stochastic*:
in general, every sample $\mathbf{z} = (z_1, z_2, \dots)$ is different
 - ⇒ DPs map a *base distribution* P_0 to a *distribution over distributions* $\text{DP}(\alpha, P_0)$
- *Pitman-Yor Processes* (PYPs) generalise Dirichlet Processes
- An *adaptor* is a function that *maps a base distribution* P_0 to a *distribution over distributions* with the same support as P_0
 - ▶ Dirichlet Processes and Pitman-Yor Processes can be used as adaptors

Adaptor grammars as generalised PCFGs

- An *adaptor grammar* is a PCFG with a set $\mathcal{A} \subseteq \mathcal{N}$ of *adapted nonterminals*, and *adaptors* C_X for each $X \in \mathcal{A}$
- *Dirichlet Process Adaptor Grammar*:
 - ▶ If $X \in \mathcal{W}$ then $G_X(X) = 1$ (all mass on singleton tree X)
 - ▶ If $X \in \mathcal{N} \setminus \mathcal{A}$ is *not adapted* then X expands as in PCFG, i.e.,:

$$G_X = \sum_{X \rightarrow Y_1 \dots Y_m \in R_X} \theta_{X \rightarrow Y_1 \dots Y_m} \text{TD}_X(G_{Y_1}, \dots, G_{Y_m})$$

- ▶ If $X \in \mathcal{A}$ is *adapted*, then PCFG distribution is adapted:

$$G_X \sim \text{DP}(\alpha, H_X)$$

$$H_X = \sum_{X \rightarrow Y_1 \dots Y_m \in R_X} \theta_{X \rightarrow Y_1 \dots Y_m} \text{TD}_X(G_{Y_1}, \dots, G_{Y_m})$$

- Other kinds of adaptor grammars use different adaptors
 - ▶ *Pitman-Yor adaptor grammars* use Pitman-Yor Processes as adaptors

Predictive distribution of DP adaptor grammars

- Predictive distribution: predict next tree t_{n+1} given previously generated trees $\mathbf{t} = (t_1, \dots, t_n)$

Predictive distribution of DP adaptor grammars

- Predictive distribution: predict next tree t_{n+1} given previously generated trees $\mathbf{t} = (t_1, \dots, t_n)$
- *Predictive model “caches” adapted subtrees:*

Predictive distribution of DP adaptor grammars

- Predictive distribution: predict next tree t_{n+1} given previously generated trees $\mathbf{t} = (t_1, \dots, t_n)$
- *Predictive model “caches” adapted subtrees:*
 - ▶ An *unadapted nonterminal* B expands using $B \rightarrow \beta$ with probability $\theta_{B \rightarrow \beta}$

Predictive distribution of DP adaptor grammars

- Predictive distribution: predict next tree t_{n+1} given previously generated trees $\mathbf{t} = (t_1, \dots, t_n)$
- *Predictive model “caches” adapted subtrees:*
 - ▶ An *unadapted nonterminal* B expands using $B \rightarrow \beta$ with probability $\theta_{B \rightarrow \beta}$
 - ▶ Each adapted nonterminal B is associated with a DP that caches previously generated subtrees in \mathcal{T}_B

Predictive distribution of DP adaptor grammars

- Predictive distribution: predict next tree t_{n+1} given previously generated trees $\mathbf{t} = (t_1, \dots, t_n)$
- *Predictive model “caches” adapted subtrees:*
 - ▶ An *unadapted nonterminal* B expands using $B \rightarrow \beta$ with probability $\theta_{B \rightarrow \beta}$
 - ▶ Each adapted nonterminal B is associated with a DP that caches previously generated subtrees in \mathcal{T}_B
 - ▶ An *adapted nonterminal* B expands:

Predictive distribution of DP adaptor grammars

- Predictive distribution: predict next tree t_{n+1} given previously generated trees $\mathbf{t} = (t_1, \dots, t_n)$
- *Predictive model “caches” adapted subtrees:*
 - ▶ An *unadapted nonterminal* B expands using $B \rightarrow \beta$ with probability $\theta_{B \rightarrow \beta}$
 - ▶ Each adapted nonterminal B is associated with a DP that caches previously generated subtrees in \mathcal{T}_B
 - ▶ An *adapted nonterminal* B expands:
 - to a subtree $t' \in \mathcal{T}_B$ probability proportional to the number of times t' was previously generated

Predictive distribution of DP adaptor grammars

- Predictive distribution: predict next tree t_{n+1} given previously generated trees $\mathbf{t} = (t_1, \dots, t_n)$
- *Predictive model “caches” adapted subtrees:*
 - ▶ An *unadapted nonterminal* B expands using $B \rightarrow \beta$ with probability $\theta_{B \rightarrow \beta}$
 - ▶ Each adapted nonterminal B is associated with a DP that caches previously generated subtrees in \mathcal{T}_B
 - ▶ An *adapted nonterminal* B expands:
 - to a subtree $t' \in \mathcal{T}_B$ probability proportional to the number of times t' was previously generated
 - using $B \rightarrow \beta$ with probability proportional to $\alpha \theta_{B \rightarrow \beta}$

Adaptor grammars for word segmentation

- Input: phoneme sequences with *sentence boundaries* (Brent)
- Task: identify *word boundaries*, and hence words

y u w a n t t u s i D 6 b U k

Words → Word

Words → Word Words

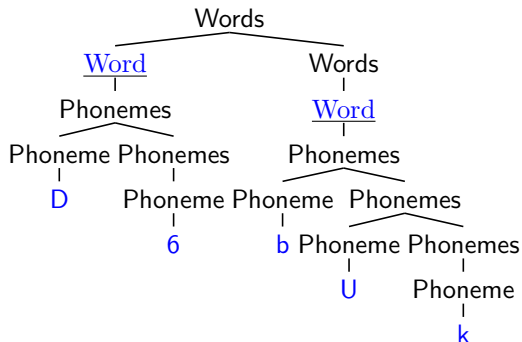
Word → Phonemes

Phonemes → Phoneme Phonemes

Phonemes → Phoneme

Phoneme → a | ... | z

- Adapted nonterminals (e.g., Word) highlighted and underlined



Adaptor grammars for named entity parsing

$NP \rightarrow (A0) (A1) \dots (A6)$ $NP \rightarrow (B0) (B1) \dots (B6)$

A0 \rightarrow Words

B0 \rightarrow Words

...

...

A6 \rightarrow Words

B6 \rightarrow Words

$NP \rightarrow$ Unordered+

Unordered \rightarrow Words

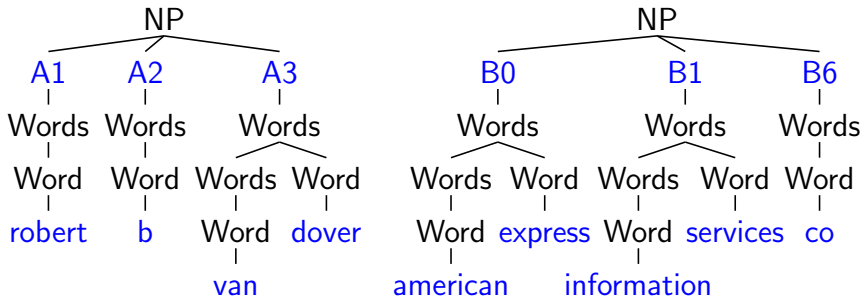
Words \rightarrow Word

Words \rightarrow Words Word

Word $\rightarrow w$ for each $w \in \mathcal{W}$

- Grammar learns three kinds of named entities
- Two are ordered sequences as before
 - ▶ each “slot” can be filled with a *collocation*
 - ▶ captures multi-word expressions like *van Dover*

Sample parses for named entities



See Elsnér, Charniak and Johnson (2009)

Outline

Topic models

Context-free grammars

Mixture topic models as PCFGs

Unsupervised learning of internal structure of named entities

Adaptor grammars: a Bayesian non-parametric extension to CFGs

Admixture topic models as PCFGs

Finding topical collocations with adaptor grammars

Conclusion and future work

Using “document ids” to identify documents

- Admixture topic models are standardly applied to entire documents
- Standard PCFG parsing algorithms require time proportional to *cube* of sentence length
 - ▶ while it's possible for a PCFG to generate full documents, with standard parsing algorithms it would be unacceptably slow
- *Document ids* permit us to break a document into several smaller chunks
 - ▶ a document id is a special nonterminal identifying the document this input came from

Admixture topic models as PCFGs (1)

- Prefix strings from document j with a *document identifier* “ $_j$ ”

Sentence \rightarrow Doc' $_j$ $j \in 1, \dots, m$

Doc' $_j \rightarrow$ $_j$ $j \in 1, \dots, m$

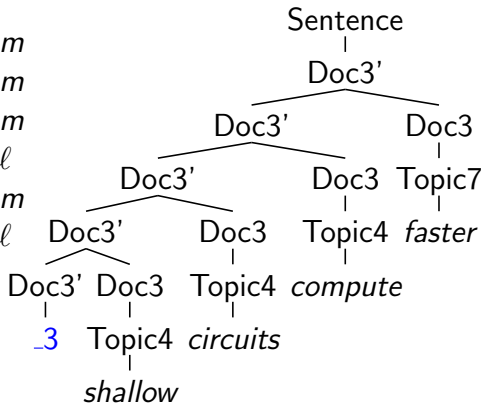
Doc' $_j \rightarrow$ Doc' $_i$ Doc $_j$ $j \in 1, \dots, m$

Doc $_j \rightarrow$ Topic $_i$ $i \in 1, \dots, l$

Topic $_i \rightarrow$ w $j \in 1, \dots, m$

Topic $_i \rightarrow$ w $i \in 1, \dots, l$

Topic $_i \rightarrow$ w $w \in \mathcal{W}$



Admixture topic models as PCFGs (2)

- Spine deterministically *propagates document id up through tree*

Sentence \rightarrow Doc'_j $j \in 1, \dots, m$

Doc'_j \rightarrow j $j \in 1, \dots, m$

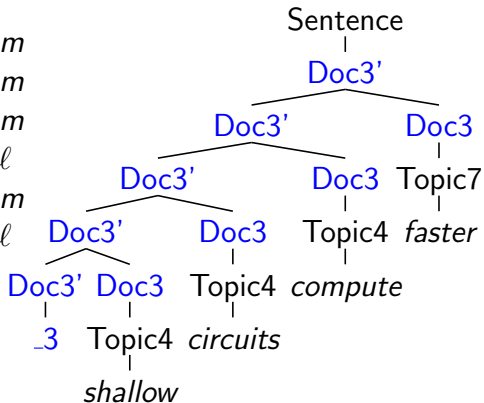
Doc'_j \rightarrow Doc'_j Doc_j $j \in 1, \dots, m$

Doc_j \rightarrow Topic_i $i \in 1, \dots, l$

Topic_i \rightarrow w $j \in 1, \dots, m$

Topic_i \rightarrow w $i \in 1, \dots, l$

Topic_i \rightarrow w $w \in \mathcal{W}$



Admixture topic models as PCFGs (3)

- $\text{Doc}_j \rightarrow \text{Topic}_i$ rules nondeterministically map *documents to topics*

$\text{Sentence} \rightarrow \text{Doc}'_j \quad j \in 1, \dots, m$

$\text{Doc}'_j \rightarrow _j \quad j \in 1, \dots, m$

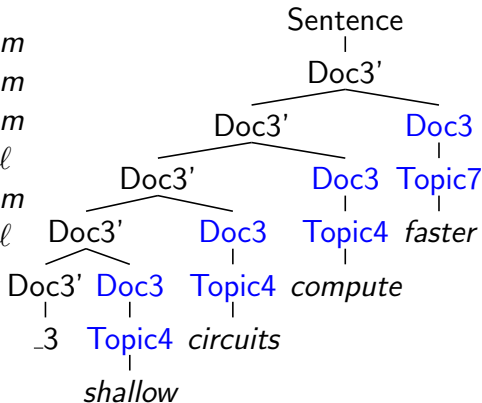
$\text{Doc}'_j \rightarrow \text{Doc}'_j \text{ Doc}_j \quad j \in 1, \dots, m$

$\text{Doc}_j \rightarrow \text{Topic}_i \quad i \in 1, \dots, l$

$\quad \quad \quad j \in 1, \dots, m$

$\text{Topic}_i \rightarrow w \quad i \in 1, \dots, l$

$w \in \mathcal{W}$



Admixture topic models as PCFGs (4)

- $\text{Topic}_i \rightarrow w$ rules nondeterministically map *topics to words*

$\text{Sentence} \rightarrow \text{Doc}'_j \quad j \in 1, \dots, m$

$\text{Doc}'_j \rightarrow _j \quad j \in 1, \dots, m$

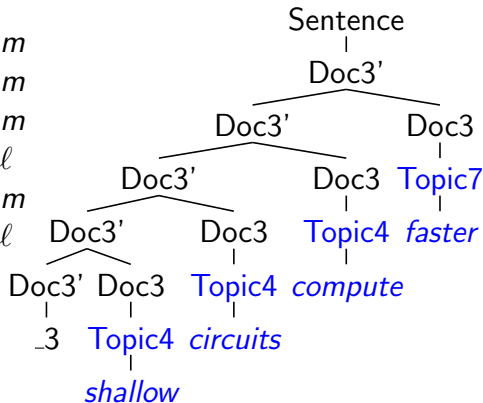
$\text{Doc}'_j \rightarrow \text{Doc}'_i \text{ Doc}_j \quad j \in 1, \dots, m$

$\text{Doc}_j \rightarrow \text{Topic}_i \quad i \in 1, \dots, l$

$\text{Doc}_j \rightarrow \text{Topic}_i \quad j \in 1, \dots, m$

$\text{Topic}_i \rightarrow w \quad i \in 1, \dots, l$

$w \in \mathcal{W}$



Outline

Topic models

Context-free grammars

Mixture topic models as PCFGs

Unsupervised learning of internal structure of named entities

Adaptor grammars: a Bayesian non-parametric extension to CFGs

Admixture topic models as PCFGs

Finding topical collocations with adaptor grammars

Conclusion and future work

Topic model with collocations

- Combines *PCFG for admixture topic model* and *segmentation adaptor grammar*

Sentence \rightarrow Doc_{*j*} $j \in 1, \dots, m$

Doc_{*j*} \rightarrow $-j$ $j \in 1, \dots, m$

Doc_{*j*} \rightarrow Doc_{*j*} Topic_{*i*} $i \in 1, \dots, l;$

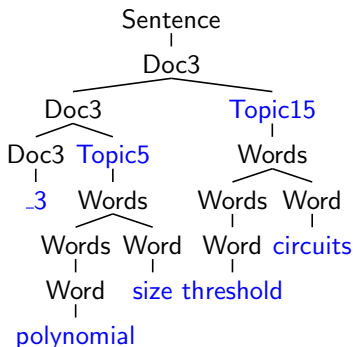
$j \in 1, \dots, m$

Topic_{*i*} \rightarrow Words $i \in 1, \dots, l$

Words \rightarrow Word

Words \rightarrow Words Word

Word $\rightarrow w$ $w \in \mathcal{W}$



Data preparation in Griffiths et al (2007)

- Documents are papers from NIPS proceedings (~ 3 million words)
- Case normalised
- Segmented at *punctuation* and *function words*

annotating an unlabeled dataset is one of the bottlenecks in using supervised learning to build good predictive models. getting a dataset labeled by experts can be expensive and time consuming. with the advent of crowdsourcing services ...

the task of recovering intrinsic images is to separate a given input image into its material-dependent properties, known as reflectance or albedo, and its light-dependent properties, such as shading, shadows, specular highlights, ...

Finding topical collocations in NIPS abstracts

- Run topical collocation adaptor grammar on NIPS corpus
- Run with $\ell = 20$ topics (i.e., 20 distinct Topic_i nonterminals)
- Corpus is segmented by punctuation
 - ▶ terminal strings are fairly short
 - ⇒ inference is fairly efficient
- Used Pitman-Yor adaptors
 - ▶ sampled Pitman-Yor a and b parameters
 - ▶ flat and “vague Gamma” priors on Pitman-Yor a and b parameters
- See Griffiths et al (2007) for an alternative topical collocation model, and Johnson and Goldwater (2009) for details on inference

Sample output on NIPS corpus, 20 topics

- Multiword subtrees learned by adaptor grammar:

T_0 → gradient descent	T_1 → associative memory
T_0 → cost function	T_1 → standard deviation
T_0 → fixed point	T_1 → randomly chosen
T_0 → learning rates	T_1 → hamming distance
T_3 → membrane potential	T_10 → ocular dominance
T_3 → action potentials	T_10 → visual field
T_3 → visual system	T_10 → nervous system
T_3 → primary visual cortex	T_10 → action potential
- Sample skeletal parses:
 - _3 (T_5 polynomial size) (T_15 threshold circuits)
 - _4 (T_11 studied) (T_19 pattern recognition algorithms)
 - _4 (T_2 feedforward neural network) (T_1 implements)
 - _5 (T_11 single) (T_10 ocular dominance stripe) (T_12 low)
(T_3 ocularity) (T_12 drift rate)

Some collocations found in NIPS corpus

Count	Topic	Collocation
2	T0	unites states israeli binational science foundation bsf
2	T5	batch k-means empty circles online gradient
12	T1	partially observable markov decision processes
12	T2	defense advanced research projects agency
7	T5	radial basis function rbf network
5	T6	analog vlsi neural network chip
4	T12	national science foundation graduate fellowship
3	T10	globally optimal on-line learning rules
3	T12	radial basis function rbf units
3	T13	non-parametric multi-scale statistical image model
3	T15	weight vector estimate requires knowledge
3	T17	orientation bands intersect ocular dominance
3	T18	optimal brain damage le cun
3	T6	normalized mean squared prediction error
47	T5	markov chain monte carlo
43	T12	radial basis function rbf
41	T12	radial basis function networks
39	T7	independent component analysis ica
35	T11	principal component analysis pca



Some collocations found in NIPS corpus (cont.)

Count	Topic	Collocation
17	T11	principal components analysis pca
16	T11	hidden markov models hmm
14	T18	artificial neural network ann
13	T15	optimal brain damage obd
12	T4	kanerva sparse distributed memory
11	T14	hybrid monte carlo method
11	T19	artificial neural networks ann
10	T0	mean square error mse
10	T12	radial basis functions rbfs
10	T16	markov decision process pomdp
10	T11	hidden markov model hmm
10	T3	atr human information processing
10	T18	artificial neural networks anns
10	T9	spin spin correlation function
9	T2	naive mean field approximation
9	T0	mean squared error mse
9	T7	support vector machines svms
9	T8	owl sound localization system
8	T1	compatible lateral bipolar transistors

Application: “perspective” and sentiment analysis

- Hardisty et al (2010) use a topical collocation model in a “perspective” sentiment analysis
- Data: the *Bitter Lemons* corpus
essays on mid-East issues from Israeli and Palestinian perspectives
- Supervised training: training sentences belong to one of two “super documents”
 - ▶ learns distributions over topics associated with each perspective
 - ▶ can be viewed as a “semi-supervised” approach
- Label test documents by finding “super document” most likely to generate them
- Compared a number of other supervised and semi-supervised methods (including SVMs, other collocation-based approaches)
- Found that *adaptor grammar topical collocations (with a hierarchical topic structure)* performed best of all

Outline

Topic models

Context-free grammars

Mixture topic models as PCFGs

Unsupervised learning of internal structure of named entities

Adaptor grammars: a Bayesian non-parametric extension to CFGs

Admixture topic models as PCFGs

Finding topical collocations with adaptor grammars

Conclusion and future work

Conclusions

- Topic models are useful for automatically classifying and extracting information from document collections
- Although PCFGs are generally thought of as methods for syntactic analysis, they can be used to express topic models as well
- Both mixture and admixture topic models can be expressed as PCFGs
- The connection between PCFGs and topic models suggests lots of new types of topic models
 - ▶ PCFGs are good at capturing structural relationships
 - ⇒ unsupervised models that learn the structure of names
- Adaptor grammars generalise PCFGs by memoising entire subtrees
 - ⇒ topic models that *learn topical collocations* rather than just topical words

Future work

- *Evaluation methods for topical collocation models*
 - ▶ many variants of these models are possible
 - ▶ each of them has tunable hyperparameters
 - ▶ application-based evaluation (e.g., Hardisty et al)
- Efficient inference procedures specialised for particular models
 - ▶ the grammar-based approach is very general
 - ▶ but more efficient procedures can be constructed for particular models
- Find other applications of topic model/grammar hybrids